IN THE CLAIMS

1-30. **(Canceled)**

31. **(New)** A method comprising:

receiving a program unit having a global storage object, wherein the global storage object is to be accessed by a thread during execution; and

adding initialization logic to the global storage object, the initialization logic to:

determine whether a cache object has been generated for the global storage object;

allocate at least one cache object for the global storage object if a cache object has not been generated; and

generate a private copy of the global storage object, wherein the private copy is retrievable by the thread directly from the at least one cache object.

32. **(New)** The method of claim 31, wherein the at least one cache object is stored in a software cache for the program unit during execution of the translated program unit.

33. **(New)** The method of claim 31, wherein the at least one cache object includes a pointer, wherein the pointer points to the private copy of the global storage object.

34. **(New)** The method of claim 33, wherein the pointer to the private copy of the global storage object is generated through execution of a routine of the run time library.

35. (**New**) The method of claim 31, wherein the pointer to the private copy of the global storage object is generated through execution of the translated program unit, independent of the run time library.

36. (**New**) The method of claim 31, wherein the program unit is translated from source code to source code.

37. (**New**) The method of claim 31, wherein the program unit is translated from source code to assembly code.

38. (**New**) The method of claim 31, wherein the program unit is translated from assembly code to source code.

39. (**New**) The method of claim 31, wherein the translated program unit is to execute in a multi-processing shared memory environment.

40. (**New**) The method of claim 31, wherein allocating the at least one cache object comprises creating the at least one cache object through an invocation of a routine within a run time library upon determining that the at least one cache object has not been allocated.

41. (**New**) A system comprising:

a translation unit to receive a program unit having a global storage object, wherein the global storage object is to be accessed by a thread during execution in a multi-processing shared memory environment, the translation unit to translate the program unit into a translated program unit, wherein the translated program unit is to determine whether a cache object has been allocated for the global storage object, to generate at least one cache object for the global storage object if a cache object has not been allocated and to generate a thread private copy of the global

storage object for the thread during execution, the thread private copy of the global storage object generated by a routine in a run time library, wherein the thread private copy of the global storage object is subsequently retrieved by the thread directly from the at least one cache object; and

a compiler unit coupled to the translation unit, the compiler unit to receive the translated program unit and to generate object code based on the translated program unit.

42. (**New**) The system of claim 41, comprising an execution unit coupled to the translation unit, the compiler unit and a run time library, the execution unit to receive the object code and to execute the object code in a multi-processing shared memory environment.

43. (**New**) A machine-readable medium that provides instructions, which when executed by a machine, cause said machine to perform operations comprising:

receiving a program unit having a global storage object, wherein the global storage object is to be accessed by a thread during execution; and

translating the program unit, wherein translating includes adding initialization logic to the global storage object, the initialization logic to include the following:

determining whether a cache object has been generated for the global storage object;

allocating at least one cache object for the global storage object if a cache object has not been generated; and

generating a private copy of the global storage object, wherein the private copy is retrievable by the thread directly from the at least one cache object.

44. (**New**) The machine-readable medium of claim 43, wherein the at least cache object includes a pointer, wherein the pointer points to the private copy of the global storage object for the thread.

45. (**New**) The machine-readable medium of claim 43, wherein the private copy of the global storage object for the thread is generated through execution of a routine of the run time library.

46. (**New**) The machine-readable medium of claim 43, wherein the private copy of the global storage object for the thread is generated through execution of translated program unit, independent of the run time library.

47. (**New**) The machine-readable medium of claim 43, wherein generating the at least one cache object during execution of the translated program unit comprises creating the at least one cache object through an invocation of a routine within a run time library upon determining that the at least one cache object has not been allocated.

48. (**New**) The machine-readable medium of claim 43, wherein the program unit is translated from source code to source code.

49. (**New**) The machine-readable medium of claim 43, wherein the program unit is translated from source code to assembly code.

50. (**New**) The machine-readable medium of claim 43, wherein the program unit is translated from assembly code to source code.

51. (**New**) The machine-readable medium of claim 43, wherein the translated program unit is to execute in a multi-processing shared memory environment.